A TOOL ASSISTING TEACHERS IN AUTOMATING THE ASSESSMENT OF PROGRAMMING ASSIGNMENTS

Avelina Fernandez[1], Jose M. del Alamo[1], Julio C. Caiza[2]

[1]Universidad Politécnica de Madrid (SPAIN)
[2]Escuela Politécnica Nacional (ECUADOR)

Abstract

Automating the assessment of programming assignments brings benefits for both students and teachers, since it helps the formers to gain a timely feedback and releases the latter from tedious tasks. The related literature in the domain has usually focused on the assessment process and the tools required for it, proposing libraries and systems that teachers can use in this process. However, few of them have work towards reducing the effort and time teachers require to properly set up new assessment processes. This paper describes our experience with the analysis and design of a new tool to support teachers in visually developing automatic graders of programming assignments, introducing the underlying concepts and technologies and presenting the system architecture.

Keywords: Programming assignment, automatic grading, evaluation metrics, assignment assessment, online assessment, Moodle, programming.

1 INTRODUCTION

The learning path in programming-related courses involves that teachers must regularly and timely assess the knowledge and practice acquired by students. For that, teachers must prepare the assignments that students must develop on their own and then submit for assessment. Two factors are closely related with the student success, namely, the number of activities they deal with and the (timely) feedback they receive for their submissions [1]. That is, the more assessments the students work in and the faster they get feedback from teachers, the better they will develop their programming skills. Although the benefits for students are clear, teachers require some tools to help them in the assessment process, otherwise unapproachable.

Some features can be assessed automatically, thus releasing teachers from this tedious task. The programming assignments usually involve text files with code and executable programs, and there are many software libraries that help in dynamically and statically assessing them. Indeed, some authors have reported their successful experience on developing and applying tools for the automatic assessment of programming [2][3][4][5][6]. These tools apply and combine different grading criteria, so as to provide students with proper and timely feedback on their submissions. However, what about the effort required for teachers to design and develop the assessment programs?

Creating, configuring and releasing an assessment task are not easy tasks. First, the teachers must look for the tools and libraries that help them with the assessment steps. Then, these tools must be properly integrated, so as to produce an assessment process tailored to each assignment. Finally, the resulting artefacts must be integrated with the learning environment in place, so that students can easily use it. As a result, automating the assessment of programming assignments emerges as a time-consuming task for most teachers, who spend many hours preparing the assessment, and then configuring and integrating it with the existing learning infrastructure.

To tackle the aforementioned issues, this paper describes our experience with the analysis and design of a tool to support teachers in developing automatic graders of programming assignments. Specifically, the tool allows teachers to combine a set of predefined grading tasks that will process the student submissions, and an easy configuration of each task. The tool provides support for ad-hoc programming assignments management, and advanced programming features such as compiling the source code, running the resulting programs, testing their functionality, or assessing the code style and documentation.

The remaining of the paper is as follows. Section 2 provides an overview of several research efforts supporting teachers in assessing computer assignments. Then, section 3 introduces the concepts of services, service oriented architecture, and mashups, which are the technological basis of the proposed solution. After that, section 4 analyses and compares existing solutions to visually compose

services. Section 5 introduces the system architecture and its major subsystems. Finally, section 6 concludes the paper.

2 RELATED WORK

Several research efforts [7] have already tapped on the issues of supporting teachers in assessing their students assignments. This section provides an overview of some of them.

CourseMarker is reported in [8]. Its architecture includes a marking subsystem which carries out the grading process, it is done taking the students' provided solution against a set of metric tools. These tools are a typographic tool, a dynamic tool (using test data) and a feature tool (check specific features). The grading process depends on a Java-written file that contains the design of the exercise and is built by a developer. This file, and therefore the process, includes calls to already developed internal tools and to externals as well. For setting the exercise's properties, a text file is used.

In [9] a tool called Marmoset is reported. It uses a client-server architecture and the server carries out the grading process, which needs of the submission and a test setup. The test setup contains all the auxiliary resources needed inside the grading process. This tool remarks its different kinds of tests that can be applied to the submission. Considering Java language, they describe the process including the use of: JUnit for running test cases, FindBugs to look for Java coding defects, and Clover to collect data from the test. The grading process is configured through a file made by a developer. This process is static and composed by a defined number of steps.

An interesting architecture based on plugins is proposed in [10]. This web tool is called WebCat (Web-based Center for Automated Testing) and aims to perform automatic grading of programming assignments. Its main features are: security, portability, extensibility, flexibility, and manual grading support as well (a detailed list of WebCAT features can be found at http://wiki.web-cat.org/WCWiki/WebCatFeatures). Every plugin is stored in the server and used inside the grading process. A usual grading process could be composed by the analysis of code correctness, test completeness, and a test-based validity. But due to its architecture, more metrics could be considered and implemented as new plugins. Eventually, the tool could consider any way to grade and could support any language that could be installed on the server. To configure the submission, it is necessary to define a set of settings, and a script which controls the grading process.

Rodríguez del Pino et al. in [11] show an interesting tool called VPL (Virtual Programming Lab) that works as a Moodle plugin. To create the task, the teaching staff has to define a set of parameters inherited from a Moodle task; but mainly it is necessary to define a set of test cases. The tool will consider the number of test cases passed. The tool has as feature to recognize automatically the programming language of a set of them supported, then a default script is used to control the grading process. To provide of more extensibility and flexibility, the evaluation script can be changed by the user.

Although these tools have their own set of highlighted features, they share something in common, and it is the use of software components inside the grading process, and necessarily the use of a file which controls that. It means that if you want to set you own grading process you can do it, but probably you will have to develop the software component and the control file.

Going a step forward, Caiza [12] made no necessary the development of a control-file, by providing an architecture to support configurable automatic grading processes. The underlying idea is to support different ways of grading source code through the use of an artifact named “grading-submodule”, which is a software component that evaluates a specific aspect of the source code by considering one metric or one criterion. The grading-submodules have to be developed by the teaching staff of by an administrator only once and then they can be reused. The structure of the grading submodule is already provided, and developers just need to extend it. The grading process configuration is defined by the teacher through a web interface where combo and text boxes aid in setting the grading submodules to be used and the needed parameters. As a result, teachers are not required to put their hands on code because the file which controls the grading process is automatically generated. Once configured and deployed, the assessment process starts whenever an assignment is submitted and it takes place in a jailed environment provided by VPL [11].

This paper builds on Caiza work, and improves it by providing a service oriented architecture and assessment development environment that enables teachers in easily generating their assessment processes, which is more convenient. For that, we have followed a service-oriented architecture,

exposing the grading submodules as services that can be easily composed. The following section provides a brief background on the underlying concepts and technologies.

3 BACKGROUND ON SERVICE-ORIENTED ARCHITECTURES

A Service-Oriented Architecture (SOA) [13] is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains [14]. These capabilities solve or support a solution for the needs that an entity, or other collaborating, face in the course of their businesses.

Services are the mechanism by which needs and capabilities are brought together. A service description contains the information necessary to interact with the service and describes this in such terms as the service inputs, outputs, and associated semantics. The service description also conveys what is accomplished when the service is invoked and the conditions for using the service.

In general, entities (people and organizations) offer capabilities and act as service providers. Those with needs who make use of services are referred to as service consumers. The service description allows prospective consumers to decide if the service is suitable for their current needs and establishes whether a consumer satisfies any requirements of the service provider.

A Web services architecture is the most common way to implement an architecture according to SOA concepts. As described by the W3C a Web service is a software system designed to support interoperable machine-to-machine interaction over a network [15]. There are many ways for a service consumer entity to engage and use a Web service. In general, the following broad steps are required:

1. the requester (consumer) and provider entities become known to each other (or at least one becomes known to the other);
2. the requester and provider entities somehow agree on the service description and semantics that will govern the interaction between the requester and provider agents;
3. the service description and semantics are realized by the requester and provider agents; and;
4. the requester and provider agents exchange messages, thus performing some task on behalf of the requester and provider entities.

Although a Web service can be built using different technologies, the two main options are SOAP (Simple Object Access Protocol) [16] and REST (Representational State Transfer) [17].

SOAP is an Extensible Markup Language (XML)-based protocol intended for exchanging structured information in a decentralized, distributed environment. It provides a standard, extensible, composable framework for packaging and exchanging XML messages. SOAP messages can be carried by a variety of network protocols such as HTTP or SMTP, or a proprietary messaging protocol.

On the other hand, the REST-based Web services are based on the Representational State Transfer architectural style (from where they take the name), and thus data and functionality are considered resources and are located using Uniform Resource Identifiers (URIs), typically Uniform Resource Locators (URLs) on the Web. The resources are acted upon by using a set of simple, well-defined operations i.e. the HTTP methods GET, PUT, POST and DELETE. REST-based Web services use a standardized data formats to exchange representations of resources e.g. XML, JSON, or Atom.

As it can be seen RESTful approaches are more focused on working with resources than messages. In addition, REST has the following advantages over SOAP: requests and responses are shorter, the bandwidth required to transport them is lesser, and the memory and processing time used while working on them is lower. Considering these advantages, and especially the feature of working more focused in interacting with resources, we decided to apply a REST approach to our solution.

Sometimes, the execution of a business process requires the composition (combination) of a set of lower-level independent building blocks of services. The logic of this process is defined by the invocations between the building blocks and how the data flow among them. The way those invocations are performed can be seen from two different points of view: orchestration and choreography [18].

In this regard, a mashup is a web-based application that integrates services from multiple sources. Some advantages of using mashups include a single data management, and the use of a single user interface to display the data, which should be as user-friendly as possible. Considering that and the

desired easiness to integrate a set of functionalities, we expose basic assessment tasks as RESTful services and orchestrate them to generate a mashup application to assess programming exercises. The following section analyses and compares existing solutions to visually compose services.

4 TOOLS FOR VISUAL COMPOSITION OF ASSESSMENTS

Our goal is aid teachers in visually building programming assignments evaluators. For that, we let them build assessment tools by visually composing a set of pre-existing grading tasks exposed as RESTful services within a SOA architecture. This section introduces the requirements for the tools aiding in this process and then analysis and compares the products current available to this end.

4.1 Requirements

In order to accomplish the main goal, the following elements are required:

- Web services that offer basic assessment tasks, e.g compilation, unit testing, among others.
- An easy-to-use tool to combine (orchestrate) the web services so as to generate an evaluation tool according to the teacher particular needs.
- A runtime environment for the services and for the evaluation tool generated.

4.2 State of the technique

Considering the requirements established previously, it is necessary to look for a tool, based on REST, to combine a set of web services. The combination could be possible in two ways, the first one requires the user to code in the languages of the involved components, and the second one could involve using only visual means.

Nowadays, there are several tools that offer an easy and intuitive development of mashup applications. In [19] a set of these tools has been studied considering criteria which include: knowledge of programming language, data for processing modules, and community support. Among others, it is possible to highlight Apatar, JackBe Presto, SnapLogic, and KapowKatalyst, because they provide great support for a wide variety of data sources, and because they can integrate other technologies and cloud services. Additionally, IBM Mashup Center, JackBe Presto and KapowKatalyst support access to other services and support the creation of graphical interfaces. These features make them powerful tools, which can be used by non-expert computer users to develop mashups. The acquisition of these tools represents a substantial financial investment that can only be satisfied by large-scale organizations, which led us to discard them for our purposes.

Netvibes (http://www.netvibes.com) and WaveMaker (http://www.wavemaker.com) are tools which not require of financial investment because they provide free access and an intuitive way to access services as well. In addition, the graphic interfaces of the applications generated are visually appealing for the final users.

Netvibes is a web service that works as a virtual desktop where you can gather all the information you want such as newspapers, blogs, weather, email, and so on. It offers a free service and a service for businesses. The free service includes monitoring tools, support for mobile, UWA (Netvibes Universal Web App), which allows the integration of virtually any tool. Additional features include: users management, editing from the mobile device, update from any place, desktop traffic analysis, and so on.

WaveMaker is a development platform for web applications and the cloud. It provides visual development tools, based on drag and drop; it allows the deployment of the application quickly and efficiently. It uses Java-based components to provide security, data access, and scalability. It also offers support for web application servers. Aditionally, it allows developers to sell and share applications, either if you use APIs provided by Wavemaker or if your entire application is original.

4.3 Comparison of mashup tools

In our study we focused on kapowKatalyst, SnapLogic, Netvibes and WaveMaker. The following table summarizes the main differences among them.

Table 1. Comparison of mashup tools.

	License	Technology	Data sources supported
KapowKatalyst	Proprietary	Java, .NET, J2EE enterprise application servers (Tomcat, IBM Websphere, Oracle WebLogic	HTML, XML, JSON, FTP, Excel, PDF, SQL Database (MS SQL Server, Oracle, IBM DB2, Sybase and MySQL), SOAP and REST Web services, SaaS, ERP, CRM Systems, Blogs, Consumer sites, Social networks
SnapLogic	Proprietary	Java, RESTful architecture	REST Web services, XML, JSON, ASN.1, Atom,HTML, Databases (DB2, MySQL, Oracle, Postgres, SQL Server), Salesforce, Netsuite, Eloqua, Coupa, SAP, Social Media (Facebook, Twitter, LinkedIn, Yelp, Foursquare, Chatter, among others), HDFS (Hadoop Distributed File Systems), Greenplum
Netvibes	Free	XHTML/XML, JavaScript/ Ajax, CSS	RSS, Atom, XML, CRM, Databases
WaveMaker	Free	Java, JavaScript	REST Web services, SOAP Web services, MySQL, PostgreSQL, HSQLDB, Oracle, Microsoft SQL Server, IBM DB2

The first criterion to filter some tools has been to look for a free tool, and thus KapowKatalyst and SnapLogic were discarded. The next criterion applied was the fact that we need a tool which allows the integration of our own developed RESTful Web services. So, we cannot use Netvibes because it only allows joining already created applications. Therefore, we will use WaveMaker, because it allows the creation of an application with services that we design and create.

Once the building blocks of our technological solution are clear, the next section introduces the system architecture and its major subsystems.

5 SYSTEM ARCHITECTURE

Considering our main goal which is aiding in the generation of tools that facilitate the automatic assessment of programming exercises, the next actors have been identified:

- Teacher, is the actor interested in building his own automatic assessment tool based on already existing services (tasks). He will also read the results of the use of these tools.
- Administrator, is the actor who is going to create a service that is associated with the basic assessment task.
- Student, is the final user who is going to work with the assessment tool to practice and get feedback or a response.

5.1 Tasks and its mapping to services

Every Web service in our system has a basic assessment task associated. This web service will be known in general as a task. Then, a task is the smallest unit of work in which you can divide the assessment tool. Every task can be reused by any flow that needs it, and besides it cannot be divided into anything smaller.

Some tasks that are required when making the correction of a programming exercise may include:

- Compiling. This task performs a set of actions including the verification that the file is written syntactically correct, a semantic verification, checking that calling to library functions is performed correctly, and finally generating a new code to communicate with the machine.
- Linking. This task refers to creating a single executable file from multiple object files.
- Running. It is the task in charge of taking the executable program and running it in a given environment.
- Testing. This task refers to running the program against a set of tests. These tests include those known as white box and black box.
    - White box tests are oriented to assess the source code and its features. This kind of test evaluates inner code of functions or methods, for instance. In this kind of test, it is possible to measure the code cyclomatic complexity.
    - Black box tests are oriented to evaluate the functionality, it means, they are oriented to evaluate the output given an input without caring of what happened inside the test or how it was done.
- Style. This task is oriented to check code standards or code conventions.
- Redundancy. This task is oriented to search for redundant code anywhere in the source code, such as recalculating a value that has been previously calculated and is still available.

Every task has to be reusable and autonomous as much as possible, which is aligned with the SOA definition of services. As a result, the next parameters are required in each service:

- A main file that is going to be evaluated by the assessment task.
- A set of secondary or auxiliary files needed to assess the main one.
- A brief description of the test result .
- An output file that will be generated if everything goes well.
- An identifier that indicates whether or not the task has successfully finished.

5.2 Sub-Systems

In order to carry out what we have explained above, we need a number of sub-systems that manage the processes involved:

- Tasks Management (services management): This component allows creating the tasks that implement the services to be visually composed. Tasks are written in Java.
- Controller: It manages the flow of execution defined for the given assessment (mashup). It displays the selected tasks, giving the necessary data, receiving and processing the result of that execution.
- Drivers Management: since they are basic services, we can create more than a flow of execution and therefore we will have to manage these flows of work.

6 CONCLUSIONS

This paper has described our experience with the analysis and design of a tool to support teachers in developing automatic graders of programming assignments by graphically combining a set of predefined grading tasks that will process the student submissions. The set of requirements and key features for these tools has been described, a comparison and analysis of supporting technologies has been carried out, and the solution architecture has been depicted.

7 ACKNOWLEDGMENT

This work has been partially supported by the Universidad Politécnica de Madrid, as part of the "*Ayudas a la Innovación Educativa y a la Mejora de la Calidad de la Enseñanza*" programme.

REFERENCES

[1] Herrera, S., San Miguel, B., Del Álamo, J.M., Cortés, M. (2012). A Proposal for Enhancing the Motivation in Students of Computer Programming. Proceedings of the 5th International Conference of Education, Research and Innovation (ICERI 2012), pp. 1157-1164.

[2] Del Álamo, J.M., Alonso, A., Cortés, M. (2012). Automated Grading and Feedback Providing Assignments Management Module. Proceedings of the 5th International Conference of Education, Research and Innovation (ICERI2012), pp. 3784-3793.

[3] Forsythe, G. E., Wirth, N. (1965). Automatic Grading Programs. Communications ACM, vol. 8, pp. 275-278.

[4] Douce, C., Livingstone, D., Orwell, J. (2005). Automatic Test-based Assessment of Programming: A Review. Journal on Educational Resources in Computing (JERIC), vol. 5, pp. 4.

[5] Ihantola, P., Ahoniemi, T.,Karavirta, V., Seppälä, O. (2010). Review of Recent Systems for Automatic Asessment of Programming Assignments. Proceedings of the 10th Koli Calling International Conference on Computing Education Research, pp. 86-93.

[6] Wang, T., Su, X., Ma, P., Wang, Y., Wang, K. (2011). Ability-training-oriented Automated Assessment in Introductory Programming Course. Computer Education, Elsevier, vol. 56, pp. 220-226.

[7] Caiza, J.C., Del Alamo, J.M. (2013). Programming assignments automatic grading: Review of tools and implementations. Proceedings of the 7th International Technology, Education and Development Conference (INTED2013), pp. 5691-5700.

[8] Higgins, C. A., Gray, G., Symeonidis, P., Tsintsifas, A. (2005). Automated Assessment and Experiences of Teaching Programming. Journal on Educational Resources in Computing (JERIC), vol. 5, pp. 5.

[9] Spacco, J., Hovemeyer, D., Pugh, W., Emad, F., Hollingsworth, J., Padua-Perez, N. (2006). Experiences with Marmoset: Designing and using an Advanced Submission and Testing System for Programming Courses. ACM SIGCSE Bulletin 38(3), pp. 13-17.

[10] Shah, A.R. (2003). Web-CAT: A Web-based Center for Automated Testing. M.S. Thesis, Dept. of Computer Science, Virginia Polytechnic Institute and State University.

[11] Rodríguez-del-Pino, J.C., Rubio-Royo, E.; Hernández-Figueroa, Z. (2012). A Virtual Programming Lab for Moodle with Automatic Assessment and Anti-Plagiarism Features. Proceedings of 2012 International Conference on e-Learning, e-Business, Enterprise Information Systems, & e-Government.

[12] Caiza, J.C. (2013). Automatic Grading of Programming Assignments: Proposal and Validation of an Architecture. M.S. Thesis, Dept. of Telematics Systems Engineering, Universidad Politécnica de Madrid.

[13] Erl, T. (2005). Service-Oriented Architecture (SOA): Concepts, Technology, and Design. Prentice Hall PTR.

[14] MacKenzie, C.M. et al (Ed.), Reference model for Service Oriented Architecture, OASIS Committee Specification 1, Aug 2006.

[15] Booth, D. et al (Ed.), Web Services Architecture, W3C Working Group Note 11 February 2004, Feb 2004; available online at http://www.w3.org/TR/ws-arch/

[16] Gudgin, M. et al (Ed.), SOAP Version 1.2 Part 1: Messaging Framework, W3C Recommendation 24 June 2003, Jun 2003; available online at http://www.w3.org/TR/soap12-part1/

[17] Fielding, R.T. (2000). Architectural Styles and the Design of Network-based Software Architectures. PhD. Dissertation. University of California, Irvine.

[18] Peltz, A. (2003). Web Services Orchestration and Choreography, Computer 36, pp 46-52.

[19] Paredes-Valverde, M. A., Alor-Hernández, G., Rodríguez-González, A., Valencia-García, R. and Jiménez-Domingo, E. (2013). A systematic review of tools, languages, and methodologies for mashup development. Software: Practice and Experience. doi: 10.1002/spe.2233